Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное  
учреждение высшего образования  
 «ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Инженерно-технологическая академия

Институт компьютерных технологий и информационной безопасности

Кафедра систем автоматизированного проектирования

Отчёт по практической работе №2

по дисциплине «Разработка серверной части веб-приложений»

**Web-приложения в Node.js**

Выполнила

студентка группы КТбо3-4 Е. О. Локота

Принял

доцент кафедры МОП ЭВМ А. Н. Шкурко

Таганрог 2023

**СОДЕРЖАНИЕ**

[**Введение** 3](#_Toc133462123)

[1 Подключение библиотек 4](#_Toc133462124)

[2 Решение задачи 4](#_Toc133462125)

[2.1 Алгоритм решения 4](#_Toc133462126)

[2.2 Программная реализация 5](#_Toc133462127)

[2.2.1 Файл index.js 5](#_Toc133462132)

[2.2.2 Файл base.liquid 7](#_Toc133462133)

[2.2.3 Файл end.liquid 8](#_Toc133462134)

[2.3 Результат работы программы 8](#_Toc133462135)

[**Заключение** 10](#_Toc133462136)

[**ПРИЛОЖЕНИЕ** 11](#_Toc133462137)

# **Введение**

**Цель работы**

Данная практическая работа направлена на отработку и закрепление навыков создания примитивных web-приложений с использованием фреймворка express.js и шаблонизатора liquid.

**Задание**

В рамках выполнения практической работы требуется выполнить следующие работы и продемонстрировать полученные навыки:

– Создать новое node.js приложение;

– Подключить к приложению библиотеки express.js и liquid;

– Разработать HTML-шаблоны для страниц приложения

– Реализовать логику обработки на сервере согласно варианту задания

**Вариант задания:**

**Задача 1**

Необходимо реализовать простой калькулятор, выполняющий сложение, вычитание, умножение и деление двух чисел. На форме калькулятора располагаются элементы управления для ввода первого операнда, операции и второго операнда. По нажатию на кнопку «Вычислить» отображается результат операции. Вычисление должно выполняться на сервере. Выполненные операции должны записываться в файл на сервере и отображаться пользователю на отдельной странице.

1. Подключение библиотек

Процесс установки необходимых библиотек изображен на рисунках 1–2.

Изображение выглядит как текст, монитор, снимок экрана, электроника

Автоматически созданное описание

Рисунок 1 – Установка liquidjs

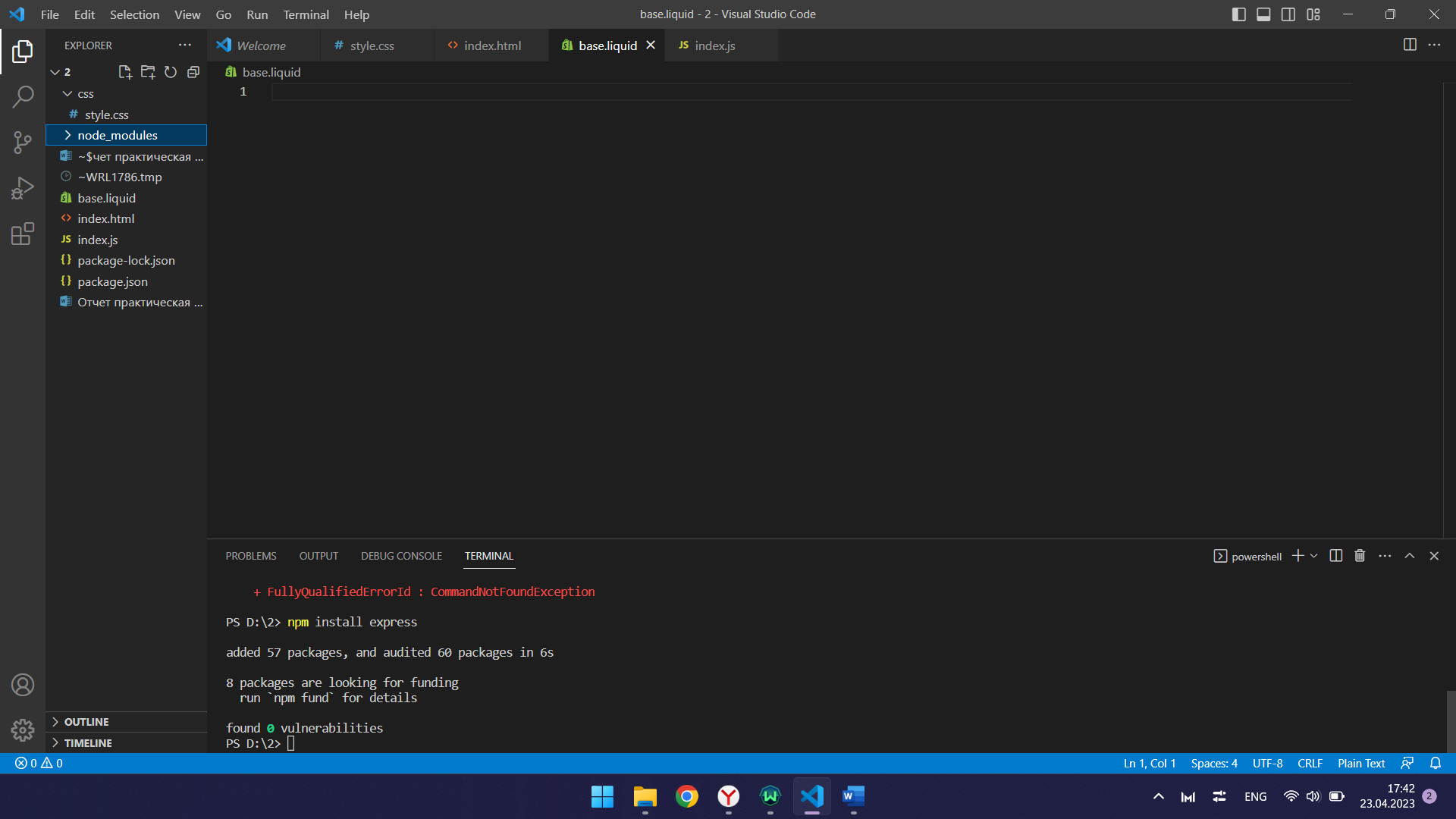


Рисунок 2 – Установка express node.js

1. Решение задачи
   1. Алгоритм решения

* Отображение основной страницы с полями для ввода, выбором знака и кнопкой.
* После ввода данных и нажатия на кнопку на странице обновляется поле «Результат» и выражение записывается в файл «results.txt»
* На основной странице присутствует кнопка «История», благодаря которой можно перейти на страницу, где изображена вся история вычислений.
  1. Программная реализация

Дерево проекта изображено на рисунке 3.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 – Дерево проекта

2. 2. 1. Файл index.js

Импортируем необходимые библиотеки (рисунок 4).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4 – Библиотеки

Далее, get-запрос, благодаря которому открывается файл base.liquid с дефолтными значениями (рисунок 5).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 – Запуск основной страницы

Благодаря post-запросу происходит обработка и вычисление. В переменную exp записывается выражение, полученное из файла base.liquid, в переменную result записывается решение, которые предоставляет функция eval(). Она считает выражение. Далее, для проверки, результат выводится на консоль (рисунок 6).

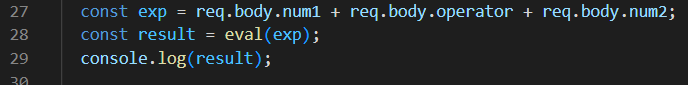


Рисунок 6 – Вычисление выражения

Далее, в файл base.liquid выводится результат и поля, которые заполнял пользователь остаются заполненными (рисунок 7).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7 – Возвращение значений

Далее, файл ‘results.txt’ дополняется выражением и после сохранения в случае, если оно произошло безошибочно, выводится соответствующее уведомление (рисунок 8).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8 – Запись выражения в файл

Если пользователь нажмет на кнопку «History», то он перейдет на новую страницу, на которой будет выведено содержимое файла.

В get-запросе читается файл ‘results.txt’, после чего данные в нем делятся по строкам и затем передаются в файл end.liquid (рисунок 9).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9 – Отображение страницы с историей вычислений

* + 1. Файл base.liquid

Данный файл состоит из двух форм, первая из которых отвечает за отображение полей для ввода, поля выбора знака и кнопки ‘Calculate’ (рисунок 10).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 10 – Форма, отвечающая за основной функционал

Вторая форма содержит в себе одну кнопку ‘History’, отчасти благодаря этой форме осуществляется переход в файл end.liquid (рисунок 11).

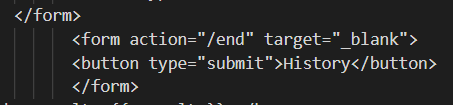


Рисунок 11 – Форма, отвечающая за одну кнопку

* + 1. Файл end.liquid

Данная форма отвечает за вывод данных из файла. Выводятся они с конца и каждый элемент с новой строки (рисунок 12).

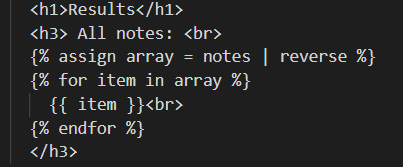


Рисунок 12 – Код для грамотного вывода

* 1. Результат работы программы

Основная страница выглядит так, как изображено на рисунке 13.

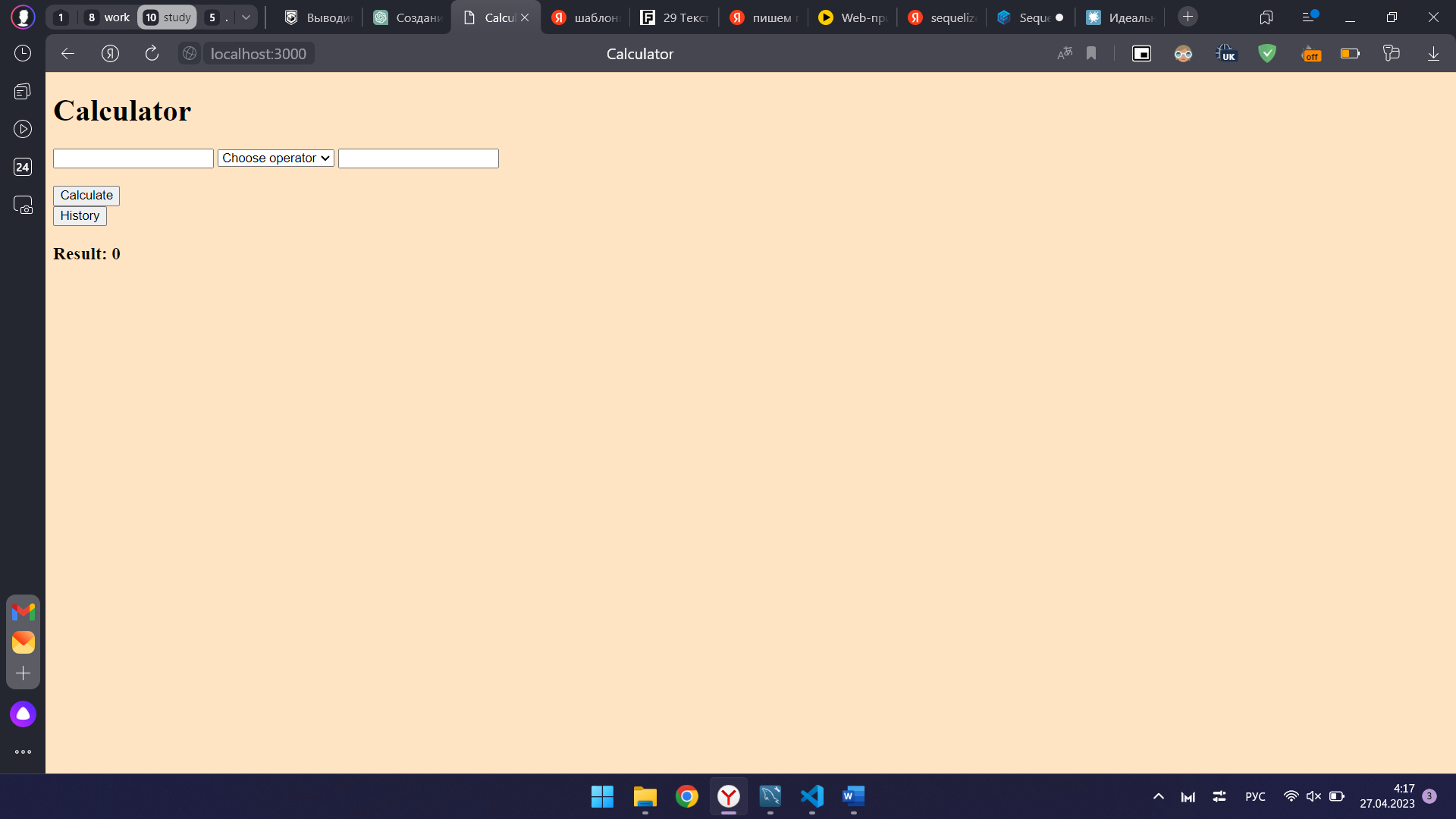


Рисунок 13 – Основная страница

После того, как пользователь ввел данные (рисунок 14), он нажимает на кнопку «Calculate» и результат обновляется (рисунок 15).

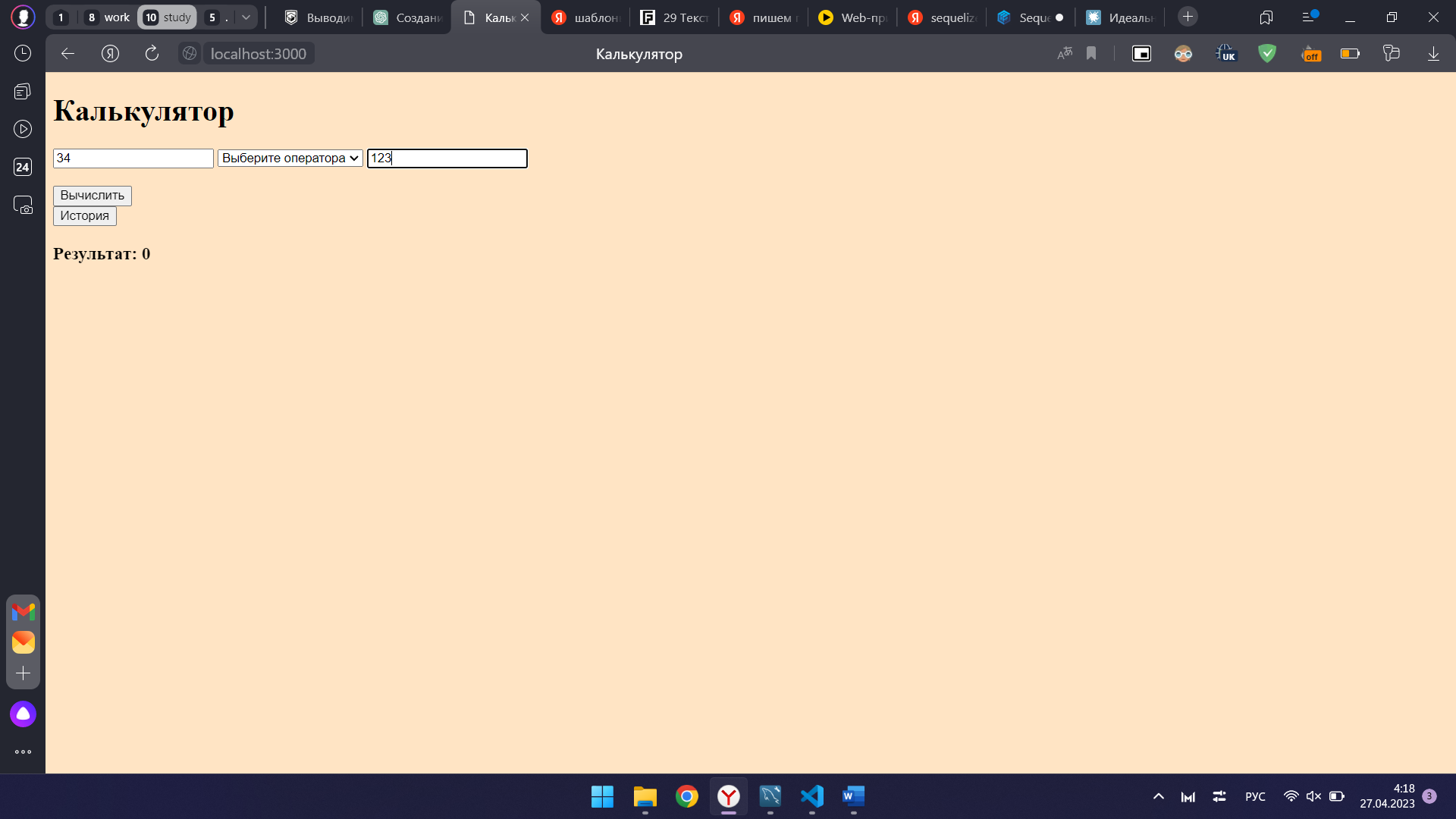


Рисунок 14 – Ввод данных

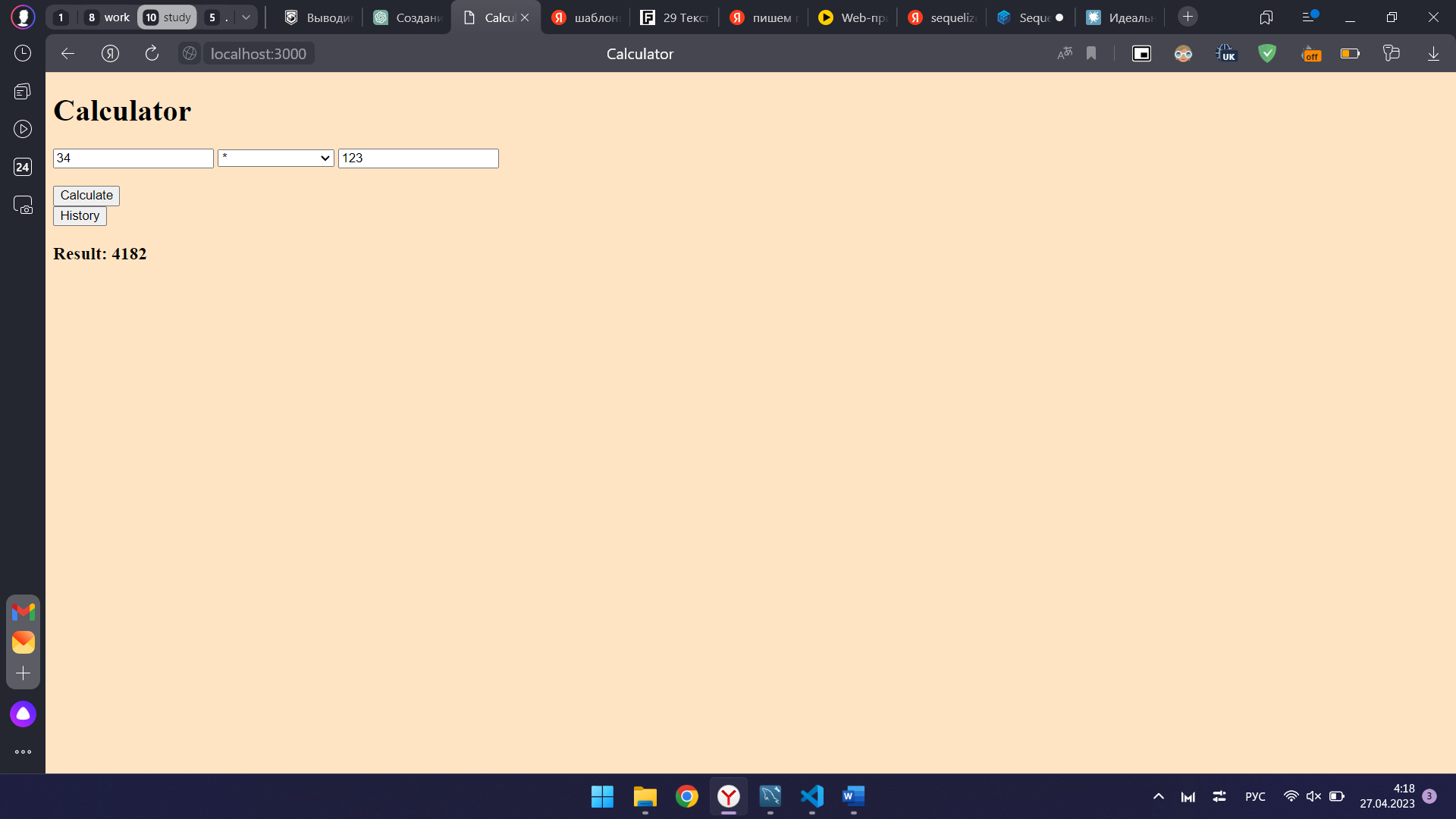


Рисунок 15 – Результат вычислений

Страница с отображением истории вычислений изображена на рисунке 16.

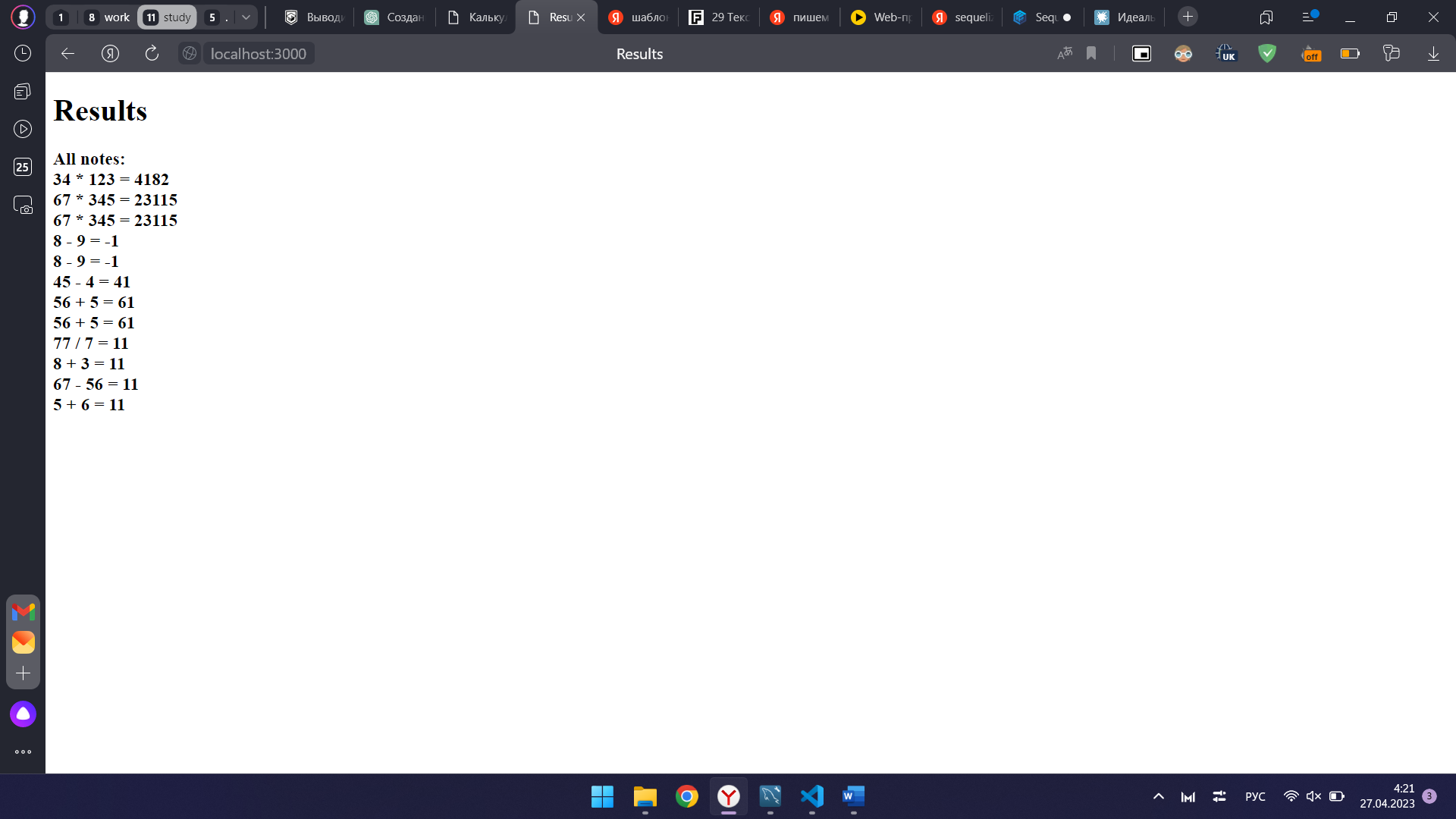


Рисунок 16 – страница с историей вычислений

# **Заключение**

В ходе выполнения практического задания была освоена работа с библиотеками express и liquidjs. Было осуществлено поставленное задание. Калькулятор выполняет свои функции, так же, страница с историей успешно отображается.

Так же, были развиты знания Java Script.

# **ПРИЛОЖЕНИЕ**

Файл index.js

const express = require('express');

const fs = require('fs');

const { Liquid }= require('liquidjs');

const app = express();

const port = 3000;

var engine = new Liquid({

  root: \_\_dirname,                                           // путь до корневой папки проекта

});

app.use(express.urlencoded({ extended: true }));

app.use(express.static('./'));

app.engine('liquid', engine.express());

app.set('views', './views');

app.set('view engine', 'liquid');

app.get('/', (req, res) => {

  res.render('base', {

    num1: '',

    num2: '',

    result: '0'

  });

});

app.post('/', (req, res) => {

  const exp = req.body.num1 + req.body.operator + req.body.num2;

  const result = eval(exp);

  console.log(result);

  res.render('base', {

    result: result,

    num1: req.body.num1,

    num2: req.body.num2,

    operator: req.body.operator

  });

  fs.appendFile('results.txt', `${req.body.num1} ${req.body.operator} ${req.body.num2} = ${result}\n`, (err) => {

    if (err) throw err;

    console.log('Result saved to file');

  });

  //res.redirect('/end');

});

app.get('/end', (req, res) => {

 const inputData = fs.readFileSync('results.txt', 'utf-8');

 const data = inputData.trim().split('\n');

 console.log(data);

 res.render('end', {notes: data});

});

app.listen(port, () => {

  console.log('Server started on port 3000');

});

Файл Base.liquid:

<!DOCTYPE html>

<html lang = "en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link href="/css/style.css" rel="stylesheet">

<title>Calculator</title>

</head>

<body>

<h1>Calculator</h1>

<form action="/" id="calcForm" method="POST" >

<input type="text" name="num1" value="{{num1}}">

<select name="operator" required>

<option value="">Choose operator</option>

<option value="+" {% if operator == '+' %}selected{%endif%}>+</option>

<option value="-" {% if operator == '-' %}selected{%endif%}>-</option>

<option value="\*" {% if operator == '\*' %}selected{%endif%}>\*</option>

<option value="/" {% if operator == '/' %}selected{%endif%}>/</option>

</select>

<input class="form\_\_field" type="text" name="num2" value="{{num2}}">

<br/>

<br/>

<button type="submit">Calculate</button>

</form>

<form action="/end" target="\_blank">

<button type="submit">History</button>

</form>

<h3>Result: {{ result }} </h3>

</body>

</html>

Файл End.liquid:

<!DOCTYPE html>

<html lang = "en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Results</title>

</head>

<body>

<h1>Results</h1>

<h3> All notes: <br>

{% assign array = notes | reverse %}

{% for item in array %}

{{ item }}<br>

{% endfor %}

</h3>

</body>

</html>